Dynamic Motion Planning of Autonomous Off-Road Vehicles

Moez Cherif

▶ **To cite this version:**

Moez Cherif. Dynamic Motion Planning of Autonomous Off-Road Vehicles. RR-2370, INRIA. 1994. inria-00074306

HAL Id: inria-00074306
https://hal.inria.fr/inria-00074306

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Dynamic Motion Planning of Autonomous Off-Road Vehicles

Moëz CHERIF

N° 2370

Octobre 1994

PROGRAMME 4

Robotique,

image

et vision

ISSN 0249-6399

Dynamic Motion Planning of Autonomous Off-Road Vehicles

Moëz CHERIF *

Programme 4 — Robotique, image et vision
Projet SHARP

Rapport de recherche n° 2370 — Octobre 1994 — 26 pages

Abstract: We describe in this paper a new motion planning approach for a non-holonomic mobile robot moving on a uneven terrain and subject to strong physical interaction constraints. The novelty of this method is that it deals with the dynamics of the robot and its physical interactions with the terrain at the planning time, together with the kinematic and the geometric constraints of the task. The planner basically combines a *geometry based reasoning strategy* operating on a subset of the configuration space of the robot, and a *local continuous motion generation technique* based on the use of a physical model of the task. We will describe each level and its corresponding models, and present how they are integrated in order to find safe and executable motions for a rover.

Key-words: Motion planning, dynamics, physical models, non-holonomic robots, off-road vehicles.

(Résumé : tsvp)

*E-mail : Moez.Cherif@imag.fr

Unité de recherche INRIA Rhône-Alpes
46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1 (France)
Téléphone : (33) 76 57 47 77 – Télécopie : (33) 76 57 47 54

Planification de mouvements dynamiques pour un véhicule autonome tout-terrain

Résumé : Nous décrivons dans ce papier une nouvelle approche pour la planification de mouvement d'un robot mobile non holonome évoluant sur un terrain accidenté et soumis à d'importantes contraintes d'interactions physiques. La nouveauté dans cette méthode est la prise en compte, en plus des contraintes géométriques et cinématiques de la tâche, de la dynamique du robot et de ses interactions avec le terrain pendant le processus de planification. Le planificateur est basé sur un processus à deux niveaux. Le premier niveau considère une formulation purement géométrique du problème de planification et opère sur un sous ensemble de l'espace des configurations du robot. Le deuxième niveau est basé sur l'utilisation d'un modèle physique de la tâche et consiste à déterminer une trajectoire exécutable par le robot tout en tenant compte des aspects dynamiques de la tâche. Nous détaillerons les deux niveaux de raisonnement utilisés ainsi que les modèles correspondants, et nous présenterons leur combinaison et intégration dans un processus de planification de mouvements d'un robot tout-terrain.

Mots-clé : Planification de mouvement, dynamique, modèles physiques, robots non holonomes, véhicules tout-terrain.

1 Introduction

The behavior of a wheeled robot moving in a natural environment depends on the combination of various geometric and physical criteria: the mechanical architecture of the vehicle, the interactions between its wheels and the terrain, the characteristics of the strategic orders, and the motion control law which are applied. Indeed, kinematics and dynamics together with phenomena such as friction, sliding or skidding play a major role in searching executable motions to achieve a task. In this paper, we address the problem of planning the motions of a non-holonomic all-terrain vehicle when dealing with parameters depending on the above-mentioned features. The method we propose basically combines two levels consisting of a *discrete search strategy* operating on a subset of the configuration space of the robot, and a *local continuous motion generation technique* based on the use of a physical model of the task.

1.1 The Problem

Let $\mathcal{A}$ be the robot, $\mathcal{T}$ be the terrain, and Q_{start} and Q_{goal} be respectively the initial and the final configurations of $\mathcal{A}$. We will denote in the sequel the workspace by $\mathcal{W}$, the configuration space of $\mathcal{A}$ by $\mathcal{CS}_{\mathcal{A}}$, and its state space by $\mathcal{SS}_{\mathcal{A}}$. Let $\mathcal{SC}$ and $\mathcal{EC}$ be respectively the set of geometric constraints and the set of kinematic and dynamic constraints to be satisfied during the execution of the task. $\mathcal{SC}$ is formulated in $\mathcal{CS}_{\mathcal{A}}$; it includes classical non-collision and contact relation constraints. These last constraints express the fact that contacts between several wheels of $\mathcal{A}$ and the terrain have to be maintained. This will allow to allow cases of tip-over of $\mathcal{A}$. $\mathcal{EC}$ is generally expressed in $\mathcal{SS}_{\mathcal{A}}$; it includes constraints resulting from the non-holonomy and the dynamics of $\mathcal{A}$, and the constraints imposed by the set of forces and torques created by both the vehicle/terrain interactions and the applied control strategy.

The problem to solve consists in finding a *safe* and *executable* continuous motion $\Gamma(Q_{start}, Q_{goal})$ and the corresponding sequence of controls U allowing to move $\mathcal{A}$ from Q_{start} to Q_{goal} while respecting the constraints of both $\mathcal{SC}$ and $\mathcal{EC}$. A motion is considered to be safe if it satisfies the constraints of $\mathcal{SC}$, and executable if it satisfies the constraints of $\mathcal{SC}$. As we will present further down, the problem of coping with the dynamic constraints of $\mathcal{EC}$ when searching for

$\Gamma(Q_{start}, Q_{goal})$ is solved by using an appropriate physical model of the task allowing the processing of the dynamics of $\mathcal{A}$ and its interactions with $\mathcal{T}$.

1.2 Related Works

The problem of planning paths for a robot moving on a planar area and subject to non-holonomic constraints has been addressed by many authors during the last years. Several techniques have been proposed to solve for such paths in different cases depending on the presence of obstacles [1][12][14][15] (see [11] for a survey of the problem). Few results have been obtained when additional dynamic constraints have to be processed, or when the robot has to move in a uneven environment [4][7]. However, some interesting contributions addressing some instances of the motion planning problem for a non-holonomic vehicle moving on hilly terrains have been reported ([3][16] and [17]).

In the approach proposed in [16], the optimal time motion planning problem is solved using an iterative two-stage process which determines a nominal path considering the shape of the terrain, and optimizes the trajectory on each of its involved patches. The main feature of this method is to consider both kinematic and simple dynamic constraints during the trajectory generation step. This is done by searching a motion constrained by a velocity curve representing the bounds of the speed of the robot along the considered nominal path. A second approach consists in finding a safe path by applying a discrete search technique through a subset of the configuration space of the robot, considering the non-holonomic constraint and a discretized model of the surface of the terrain [17]. No dynamics is taken into account during the planning process. This is reduced to determining a sequence of configurations satisfying the static equilibrium of the robot on the terrain. The considered robot/ground interactions, as also in [16], are mainly restricted to simple criteria based on the formulation of no-sliding, no tip-over and no-collision constraints.

Despite the solutions provided by these techniques, such assumptions may be insufficient to generate executable motions since the behavior of the robot strongly depends on its dynamics and on the effects of the physical interactions with the terrain. The main idea of our approach is close to that proposed in [17] but extended to cope with dynamics and physical interactions constraints.

2 The Approach

The vehicle considered in this paper is an articulated non-holonomic vehicle having a locomotion system composed of three axles each having two motorized wheels. The rear and front axles are connected to the main body of the robot by revolute joint mechanisms allowing them to have roll and pitch movements (see Figure 1). The purpose of such mechanisms is to allow $\mathcal{A}$ to be constantly in contact with the ground and to adapt its configuration to the irregularities of the terrain. However, this leads $\mathcal{CS}_{\mathcal{A}}$, and consequently $\mathcal{SS}_{\mathcal{A}}$, to be of a high dimension since a full configuration Q of $\mathcal{A}$ is given by $6 + n_\delta$ parameters: six parameters $(x, y, z, \theta, \varphi, \psi)$ specifying the position/orientation (yaw,roll and pitch) of the main body in the reference frame of the workspace $\mathcal{W}$, and n_δ parameters specifying the values of the joints corresponding to the axles and the wheels.

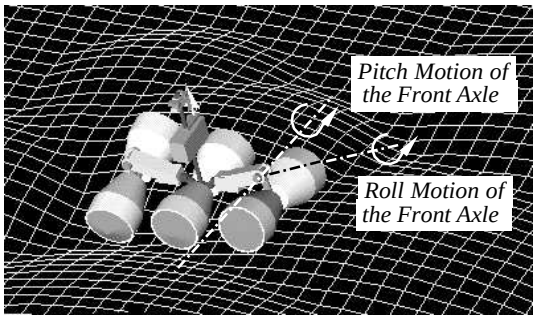

Figure 1: The six-wheeled robot $\mathcal{A}$.

Besides, dealing with dynamics and physical vehicle/terrain interactions when solving the global motion planning problem (between Q_{start} and Q_{goal}) may lead to heavy computational burden. This is due to the fact that we have to operate in the state space $\mathcal{SS}_{\mathcal{A}}$ in order to cope with both complex differential equations depending on the constraints of $\mathcal{EC}$ and more classical constraints of $\mathcal{SC}$. In order to make the addressed problem more tractable, we propose a two-level planning method based on the following features:

A. Discrete Searching of a subset of $\mathcal{CS}_{\mathcal{A}}$ As mentioned earlier, only the configurations Q of $\mathcal{A}$ satisfying the constraints of $\mathcal{SC}$ are considered. In that case, the parameters of Q are not independent and the inter-relations between them depend on the vehicle/terrain relation (i.e. distribution of the contact points according to the geometry of the terrain). A practical consequence of this property is that it allows to reduce the search space into a reduced subset $\mathcal{C}_{search}$ of $\mathcal{CS}_{\mathcal{A}}$ defined on (x, y, θ) —the horizontal position and the heading angle of the main body of $\mathcal{A}$ denoted from now by q. Afterwards, the solution is iteratively computed between q_{start} and q_{goal} corresponding to Q_{start} and Q_{goal}, by applying a discrete search technique (an A^* algorithm, for instance) through an incrementally generated directed graph $\mathcal{G}$ representing the explored configurations of $\mathcal{C}_{search}$. Such an approach has already been used in [1] and [17] to find non-holonomic paths for a robot moving on planar and $3D$ areas, respectively.

B. Solving locally continuous motions of $\mathcal{A}$ under $\mathcal{EC}$ constraints Since the generation of the successors of a node $N(q)$ of $\mathcal{G}$ does not account either the geometric shape of $\mathcal{T}$ or the dynamics of the task, we process the second level in order to cope with such features. This consists of computing locally a continuous motion of $\mathcal{A}$ satisfying the constraints of $\mathcal{EC}$ between q and q_{next}. q_{next} corresponds to the configuration of the best successor node of $N(q)$ when searching $\mathcal{G}$. This is solved by formulating the planning problem in $\mathcal{SS}_{\mathcal{A}}$ and using a physical model of the task.

Unlike methods operating in two stages: processing a geometric path, and afterwards generation of a full trajectory of the robot when considering the task execution constraints, our approach allows to introduce dynamics and vehicle/terrain interactions at the planning time. Indeed the exploration of $\mathcal{C}_{search}$ is mainly used to guide the search process, and to give us only potential intermediate configurations of $\mathcal{A}$ approximating the final solution. The real trajectory of the robot is provided by the sequence of the local motions computed when the physical model of the task are processed. The main advantage of such an approach is to cope locally with the dynamic constraints of the task as it has been already proposed in [5].

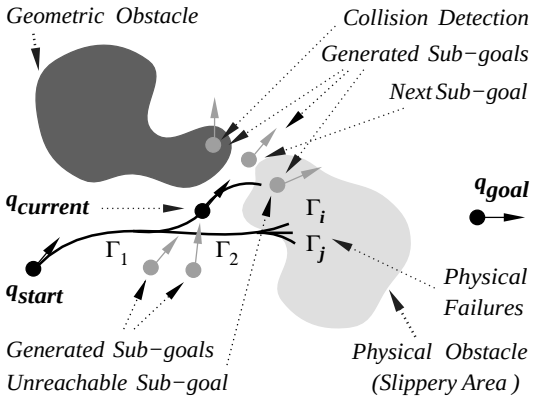

Figure 2: The general scheme of the planning approach.

At a given step of the algorithm, the potential successors of the current configuration q are explored in order to verify the safety constraints of the task (e.g. no-collision with the geometric obstacles). Afterwards, a motion $\Gamma(q, q_{next})$ is computed to move $\mathcal{A}$ from q towards its best safe neighbor q_{next}. The solution Γ is incrementally built until reaching the goal configuration q_{goal}. For instance, the motion solution Γ to be processed in figure 2 is given by the previously computed motion Γ_1 between q_{start} and q, and the concatenation of the sub-motions $\Gamma(q, q_{next})$ and $\Gamma(q_{next}, q_{goal})$ which have to be processed. If the $\Gamma(q, q_{next})$ cannot be computed because of sliding or skidding of $\mathcal{A}$ on slippery areas (such as when computing Γ_i and Γ_j), the algorithm backtracks in order to select an other potential intermediate configuration to reach and to guide the search until q_{goal}. The physical modeling of the task and the geometric and physical levels of the planning algorithm are presented in §3, §4 and §5, respectively.

3 Task Modeling

3.1 Physical Modeling of the Robot $\mathcal{A}$

In order to build the dynamic model of the robot, we have considered $\mathcal{A}$ as a network of interacting rigid objects Ω_i obeying the laws of physics of solids

connected by visco-elastic connectors [5]. This enables to couple directly each element of the robot to those others that are in contact with it. It is different from the dynamic model based on a joint space formulation where every link is related to the one immediately before it, and every motion is sensed in relative and require successive transformations processing which can be time consuming.

Let $r_i(t)$ and $\alpha_i(t)$ be, respectively, the position and the orientation parameters at time t of a given Ω_i of $\mathcal{A}$. The motions of Ω_i are specified by the Euler/Newton equations: $F_i = m_i\,\ddot{r}_i(t)$ and $T_i = \dot{L}_i(t) = I_{\Omega_i}\,\ddot{\alpha}_i(t)$, where F_i and T_i are respectively the sums of forces and torques applied on Ω_i, $L_i(t)$ is the angular momentum about the center of mass G_{Ω_i}, and I_{Ω_i} is the inertia tensor of Ω_i about the frame axes. $\dot{L}_i(t)$ is also related to Coriolis and centrifugal terms (see [8]), but we will make the assumption that such terms are negligible. Then, F_i and T_i can be computed using the *Euler*'s principle of superposition: $F_i = F_d + \sum_{force\,j} F_{i,j}$ and $T_i = U_i + \sum_{force\,j}(G_{\Omega_i}P_{i,j} \times F_{i,j})$. $T_{i,k}$ are the torques acting on Ω_i, $P_{i,j}$ are the points where the forces $F_{i,j}$ are applied, $G_{\Omega_i}P_{i,j}$ is the vector from G_{Ω_i} to $P_{i,j}$, and $\times$ is the outer product. The term F_d includes the gravity forces and additional viscous forces of the environment. The set of forces $F_{i,j}$ results from the the physical interactions of Ω_i with the other components of $\mathcal{A}$ —through the joints of $\mathcal{A}$— and with the involved components of the terrain —through the wheel/ground contact interactions. When Ω_i is a wheel, U_i corresponds to the torques generated by the control mechanisms (i.e. the "physical effector") applied on Ω_i, otherwise this term vanishes.

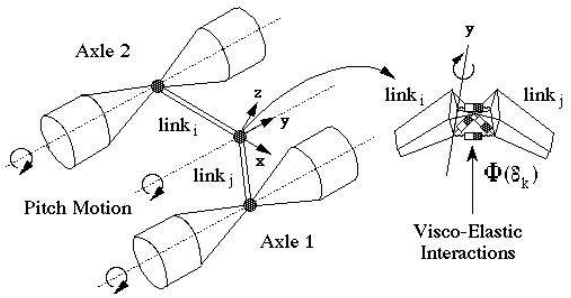

Figure 3: Modeling a $3D$ revolute joint mechanism.

Each articulated mechanism associated with the joint δ_k of $\mathcal{A}$ is represented by a network $\Phi(\delta_k)$ combining a set of connectors c_r and a set of specific points selected on the rigid bodies Ω_i corresponding to the joint δ_k. The connectors are defined in terms of visco-elastic laws (i.e. combination of springs and dampers). For instance, we have represented $3D$ rotoïd joint mechanisms by two rigid bodies connected through two pairs of points respectively selected on them and belonging to the rotation axis (see Figure 3). Such a representation has been initially proposed in [10] to model a robot entirely described in terms of elementary particles and visco-elastic connectors.

3.2 Modeling the Terrain $\mathcal{T}$

The terrain $\mathcal{T}$ is initially described by a set of interconnected patches built from an elevation map, i.e. a regular grid (x, y) expressed in the reference frame of $\mathcal{W}$. Dealing with robot/terrain interactions requires to build a representation $\Phi(\mathcal{T})$ of the terrain which is able to capture both the geometric and the physical properties of $\mathcal{T}$ and which allows the formulation of such interactions. For that purpose, we have implemented a model inspired by the concept of *discrete deformable models* [13], initially proposed for computer graphics applications.

According to this concept, the terrain is represented by a set of interconnected particles $\Phi(P_i)$ having the following properties [10][13]: (1) each particle is seen as a point mass m_i which obeys Newtonian dynamics —given by the equation $F_{P_i} = m_i \ddot{r}_{P_i}$ where r_{P_i} is the position of $\Phi(P_i)$ in $\mathcal{W}$— and which is surrounded by a spherical non-penetration "elastic" area; (2) the set of particles corresponds to the inertial and spatial occupancy characteristics of the modeled area of $\mathcal{T}$; (3) the particles are interconnected using interaction components referred to as the "connectors". Each connector corresponds to a type of interaction, and is modeled using appropriate physical laws allowing several types of behaviors (e.g. viscous/elastic or elastic cohesion, dry friction interactions).

The discretization of the terrain in terms of such elementary physical components accounts several criteria such as the terrain surface shape, the average distribution of the contact points between the wheels and the ground, and the complexity of $\Phi(\mathcal{T})$ (i.e. the number of the processed particles). In the current implementation of the system, the particles distribution is determined

by computing a set of spheres S_i whose profile approximates the surface of $\mathcal{T}$ given by the set of the initial geometric patches. This is done in such a way that each point of the terrain surface should be located on the surface of at least one S_i. Afterwards, a dynamic behavior is "given" to the computed set of spheres by placing a particle $\Phi(P_i)$ at the center of each S_i (Figure 4). The main advantage of such a representation is related to the fact that it allows us to maintain the geometric features of the motion planning problem (i.e. checking the geometric constraints as the non penetration in the ground), and to uniformly process the physical behavior of $\mathcal{T}$ and its interactions with $\mathcal{A}$.

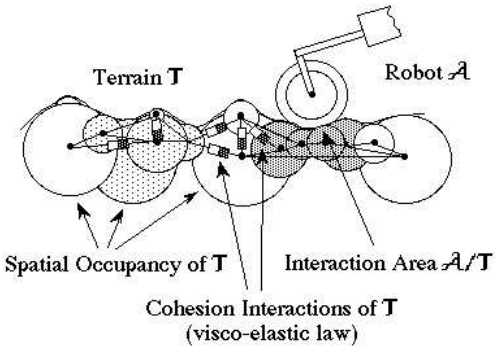

Figure 4: The Physical model of the terrain.

3.3 Coping with $\mathcal{A}/\mathcal{T}$ Interactions

The most difficult problem to solve when dealing with a vehicle moving on a terrain, is to define realistic vehicle/terrain interaction laws. Such interaction laws depend on both the distribution of the contact points and the type of the surface-surface interactions.

A. Computing the contact distribution Describing $\mathcal{T}$ in terms of spheres allows us to handle a smaller amount of information to represent its geometric shape (as shown in Figure 4). Furthermore, the combination of such simple primitives with an appropriate hierarchical description of the wheels allows us to compute easily the distribution of the contact points using a fast distance computation algorithm involving the structured sets of the considered spheres

[9]. Once a contact between the wheels of $\mathcal{A}$ and a sphere S_i corresponding to $\mathcal{T}$ is detected, the corresponding physical interaction is easily computed by activating the associated dynamic laws.

B. Processing the interactions The surface-surface interactions are processed using two types of constructions: a visco-elastic law associated with the set of S_i involved in the contact, and surface-surface interactions. In order to solve the second point, we make use of a *finite state automaton* since complex phenomena like dry friction basically involves three different states: no contact, gripping, and sliding under friction constraints. The commutation from one particular state to another is determined by conditions involving gripping forces, sliding speed, and relative distances. Each state is characterized by a specific interaction law. For instance, a viscous/elastic law between the interacting points of the wheels of $\mathcal{A}$ and $\mathcal{T}$ is associated with the gripping state, and a Coulomb equation is associated with the sliding state (see [10]).

4 The Geometric Level

4.1 The Basic Algorithm

The general algorithm basically consists of applying an A^* algorithm to find a near-optimal solution in an incrementally generated directed graph $\mathcal{G}$ representing the explored configurations of $\mathcal{C}_{search}$. In order to reduce the exploration time of the algorithm, each node $N(q)$ of $\mathcal{G}$ is defined by a neighborhood of the corresponding configurations q. During the construction of $\mathcal{G}$ which is started from the node $N(q_{start})$, two lists of nodes are handled. The first list LC contains all the generated nodes that have not been expanded[1] yet. LC is maintained sorted according to the considered cost function. The second list LEC contains all the nodes that have been checked for safety and execution constraints and eventually also expanded.

At each iteration of the algorithm, a node $N(q_i)$ which has been previously generated is selected from LC, removed to LEC, and checked if it corresponds

[1] A node N is expanded if its successors have been generated in $\mathcal{G}$, i.e. it is the start point of at least one arc of $\mathcal{G}$.

to a safe configuration. This consists in computing the "placement" relationship $Q_i = P\ell(q_i)$ providing the associated complete configuration of $\mathcal{A}$ when evaluating the constraints imposed by the wheel/ground contacts, the joints characteristics, and the stability of $\mathcal{A}$ on the terrain (see [2]). If q_i does not satisfy $\mathcal{SC}$ constraints, the node $N(q_i)$ is discarded. Otherwise, we apply the physical level of the algorithm in order to compute locally an executable motion between $q_{i,p}$ and q_i (as described in §5), where $N(q_{i,p})$ is the node which the expansion has previously provided q_i. In the case that such a motion cannot be generated, $N(q_i)$ is discarded, otherwise it is expanded. As we will present further down, this consists in generating the set of its potential successors using "simple" non-holonomic paths. A node in $\mathcal{G}$ is then associated to each resulting configuration, before being stored in the list LC. As we intend to find a continuous motion of $\mathcal{A}$, the expansion of $N(q_i)$ is actually achieved starting from the configuration $q_{i,r}$ reached after having processed the local physical motion of $\mathcal{A}$, and located in the neighborhood of q_i.

The exploration of $\mathcal{C}_{search}$ (see Figure 5) iterates until the list LC is empty, or the node $N(q_{goal})$ is reached. In this case, the trajectory solution $\Gamma(q_{start}, q_{goal})$ of $\mathcal{A}$ is provided by the sequence of the sub-motions Γ_i computed between the set of intermediate nodes of $\mathcal{G}$ located on the best path ending at $N(q_{goal})$ and starting at $N(q_{start})$.

4.2 Expansion of a Node $N(q_i)$ of $\mathcal{G}$

Let L be the length of $\mathcal{A}$ and let the spent time correspond to 0 at the configuration q_i (i.e. $q_i = q(0)$). Assuming that $\mathcal{A}$ moves without sliding, the parameters of $\dot{q}$ at time t are written, according to the non-holonomic constraint of $\mathcal{A}$ as:

$$\dot{x}(t) = v\cos(\theta), \quad \dot{y}(t) = v\sin(\theta), \quad \dot{\theta}(t) = \frac{v}{L}\tan(\phi)$$

where ϕ is the steering angle of $\mathcal{A}$ and v is the velocity of its main body. If $\phi_{max} > 0$ and $V_{max} > 0$ are the maximum steering angle and the maximum linear velocity of $\mathcal{A}$, respectively, we have at each time $\phi = \varepsilon_\phi \phi_{max}$ and $v = \varepsilon_v V_{max}$, where $|\varepsilon_\phi| \leq 1$ and $|\varepsilon_v| \leq 1$. This means that the velocity v is upper-bounded by V_{max} and the turning radius ρ_ϕ of $\mathcal{A}$ is at each time lower-bounded by $\rho_{\phi_{max}} = L/\tan(\phi_{max})$.

By assuming that ε_ϕ and ε_v remain constant within the time interval $[0, \Delta T_G]$, and integrating the equations of $(\dot{x}, \dot{y}, \dot{\theta})$, the new configuration $q(\Delta T_G)$ of $\mathcal{A}$ is written as[2]:

$$\begin{cases} x(\Delta T_G) &= x(0) + \frac{L}{\tan(\phi)} \left(\sin(\theta(\Delta T_G)) - \sin(\theta(0))\right) \\ y(\Delta T_G) &= y(0) - \frac{L}{\tan(\phi)} \left(\cos(\theta(\Delta T_G)) - \cos(\theta(0))\right) \\ \theta(\Delta T_G) &= \theta(0) + \frac{v}{L} \tan(\phi)\ \Delta T_G \end{cases}$$

The successors of q_i are then easily computed by instantiating $q(\Delta T_G)$ with a set of discrete and time constant values of $(\varepsilon_\phi, \varepsilon_v)$ within the interval $[-1, 1]^2$. In the simulation results presented in this paper, $(\varepsilon_\phi, \varepsilon_v)$ is selected among the set $\{-1, 0, 1\} \times \{-1, 1\}$, and ΔT_G is chosen so that the distance between q_i and its successors is of the size of $\mathcal{A}$ (as in Figure 5). This allows to generate 6 types of motion: 3 forward movements satisfying the non-holonomic constraints of $\mathcal{A}$ (left turn, straight, and right turn), and 3 similar backward movements.

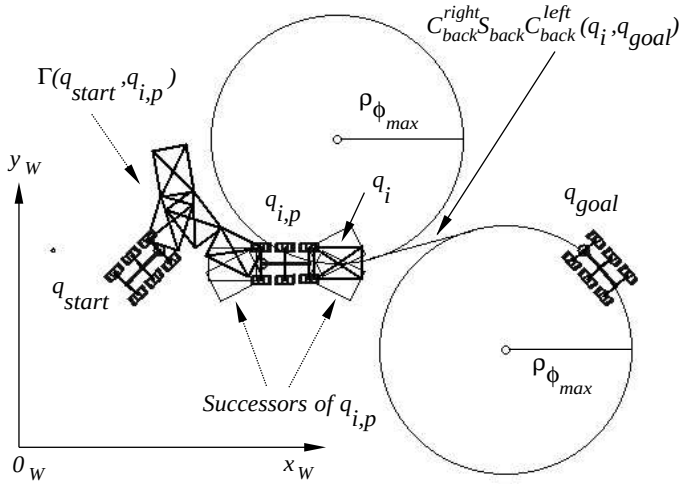

Figure 5: The geometric planning level.

[2]If $\phi = 0$, $q(\Delta T_G)$ is given by $(x(0) + v\cos(\theta(0)).\Delta T_G, y(0) + v\sin(\theta(0)).\Delta T_G, \theta(0))$.

4.3 The Cost Function

The cost function considered in this paper expresses the length of the generated trajectories of $\mathcal{A}$. For a configuration q that have been expanded (i.e. verifying the safety and execution constraints), it is given by the length of the sequence of the local motions previously computed by the physical level of the algorithm between q_{start} and q, and an heuristic term estimating the value of the remaining distance to reach q_{goal}. Using a simple and admissible heuristic based on the Euclidean distance may not in some cases be suitable for non-holonomic systems. In order to mitigate the problem, we have implemented a more informed and simple heuristic which takes into account the constraint on the minimum turning radius $\rho_{\phi_{max}}$ of $\mathcal{A}$. It consists in computing the shortest path length between the set of *geodesic* paths connecting q and q_{goal} and of the form C_1SC_2, $C_1S|C_2$, $C_1|SC_2$ or $C_1|S|C_2$. C_1 and C_2 denote two circular arcs of radius $\rho_{\phi_{max}}$ tangent to q and q_{goal} respectively, and S is a segment of a straight line tangent to C_1 and C_2 (see Figure 5). The symbol |, if it exists, indicates if the robot changes its direction at the intersecting point between one C and S. In some cases, one or more of the symbols C_1, C_2 and S may vanish (e.g. when the shortest path is of type C when q and q_{goal} have the same center of gyration, or of type S when q and q_{goal} are collinear). We have used these simple forms of paths for tractability. The problem of using complete metric for non-holonomic robots based on shortest *geodesic* curves such as Reeds-Shepp curves [15] is mainly due to the required processing time. An interesting technique has been presented in [14] allowing to use such metric via an optimized table built off-line.

5 The Physical Planning Level

5.1 The Problem

The purpose of this level is to plan a local executable motion Γ_i allowing $\mathcal{A}$ to move between two successive configurations $q_{i,p}$ and q_i when the kinematic and dynamic constraints of $\mathcal{A}$ and its physical interactions with $\mathcal{T}$ are dealt with. In order to take into account such features, we reformulate this planning problem in the state space $\mathcal{SS}_{\mathcal{A}}$ since it is required to process differential equations of

motion of $\mathcal{A}$. It is solved by searching the sequence $\mathcal{U}_i$ of bounded controls $U(t)$[3] allowing $\mathcal{A}$ to move continuously between the two states $s(q_{i,p})$ and $s(q_i)$ corresponding respectively to $q_{i,p}$ and q_i. $U(t)$ denotes the controlled torque vector applied on the physical effectors (the wheels) of $\mathcal{A}$. Since $q_{i,p}$ has been previously reached and expanded in $\mathcal{G}$, the state $s(q_{i,p})$ is completely known.

As described in §4.2, only simple non-holonomic path (circular arc or straight line) denoted by $\mathcal{P}_i^0$ and plotted on the horizontal plane is considered during the geometric generation of q_i. No relief of the terrain is taken into account. Furthermore, the velocity and the angular gyration of $\mathcal{A}$ are assumed to be constant along this path. This may be, in some cases, difficult to ensure —for instance when $\mathcal{A}$ has to cross a slope with bounded controls $U(t)$. Indeed, such assumptions require that $\mathcal{A}$ should turn or change direction instantaneously which necessitates near infinite accelerations at the turning and maneuvering points, or that it should stop, change its steering angle and then move which constrain to have a large execution time and non smooth motion. All these assumptions may be obviously restrictive to predict the complete state $s(q_i)$ of $\mathcal{A}$.

In order to estimate $s(q_i)$ and to achieve the smoothness of the global planned trajectory Γ of $\mathcal{A}$, we have defined a set of conditions on the local searched solution Γ_i and on the parameters of $s(q_i)$. These conditions depend on the fact that q_i corresponds to a maneuvering point, or located on a turning or a straight path[4]. At each time t, $\mathcal{A}$ is constrained to have a linear velocity $v(t) \leq V_{max}$. If $\mathcal{A}$ executes a straight motion, its angular velocity at each time t should be approximately zero. When q_i is a maneuvering point, $\mathcal{A}$ has to converge towards the neighborhood of q_i with a null linear velocity. In order to ensure the smoothness of Γ and to minimize the skidding of the wheels, $\mathcal{A}$ is controlled to execute the turns with a velocity upper-bounded by a positive term V_{max}^{turn}. This term depends in general on the acceleration of $\mathcal{A}$, friction and the type of interactions between $\mathcal{A}$ and $\mathcal{T}$. In the sequel, V_{max}^{turn} is given by an arbitrary value so that $V_{max}^{turn} < V_{max}$. Besides, in the cases where q_i is

[3] $U(t)$ is a 6-dimensional vector. Each element $U_r(t)$ corresponds to the r^{th} wheel of $\mathcal{A}$, and is constrained by $|U_r(t)| \leq U_{max}, \forall t$.

[4] This is determined by comparing the path $\mathcal{P}_i^0$ (of type S or C) used to generate q_i during the expansion of $q_{i,p}$ (§4.2), and the first sub-path of type C or S of the shortest *geodesic* path estimating the distance between q_i and q_{goal} (§4.3).

a maneuvering point or corresponds to a point where the steering angle ϕ has been changed (e.g. intersection point between a turn and a straight motion and vice-versa), we constrain $\mathcal{A}$ to converge towards q_i with a minimum angular velocity ($\dot{\theta}(q_i) \approx 0$). This allows to avoid the sliding and/or the skidding of the wheels when starting from q_i in the following iteration of the planning algorithm.

5.2 The Physical Planning Algorithm

We detail in this section the local planning algorithm allowing to move $\mathcal{A}$ from $s(q_{i,p})$ towards $s(q_i)$. Let consider that the state $s(0)$ of $\mathcal{A}$ at time 0 coincides with $s(q_{i,p})$. We assume that the time ΔT_{Γ_i} required to move $\mathcal{A}$ towards q_i is unknown and unbounded since we cannot establish a direct relation between it and the time increment ΔT_G used in the geometric level which does not include the execution constraints of the task. Let δt be the time increment of the trajectory planning process and corresponding to the integration time step when solving the equations of motions of $\mathcal{A}$ and $\mathcal{T}$. We denote in the sequel by $(F_{r,\mathcal{A}/T}(t), T_{r,\mathcal{A}/T}(t))$ the resultant force/torque vectors applied on the r^{th} wheel and provided by the interactions with the components of $\mathcal{A}$ linked to it (see §3.1) and the interactions with $\mathcal{T}$ (see 3.3). $\tilde{\omega}$ will refer to the angular velocity vector of the wheels required to move $\mathcal{A}$ along a given path without accounting the dynamics of the task.

The local planner is based on an iterative process involving two complementary steps. At each iteration k, the first step consists in hypothesizing a nominal sub-path $\mathcal{P}_i^k$ with no maneuvers between a configuration q_{cur} corresponding to the best state s_{cur} selected from the list of the sub-motions previously computed, and q_i. Afterwards, a motion Γ_i^k along $\mathcal{P}_i^k$ is searched while processing the physical models of $\mathcal{A}$ and $\mathcal{T}$, the interactions between them, and the execution and convergence constraints. During the computation of the solution $(\Gamma_i, \mathcal{U}_i, \Delta T_{\Gamma_i})$, a list LT of the generated sub-trajectories Γ_i^k and their associated controls $\mathcal{U}_i^k$ is maintained. The basic algorithm is described in the following and illustrated in Figure 6. It starts from $s(q_{i,p})$ when considering as initial nominal path $\mathcal{P}_i^0$ provided by the geometric level.

algorithm PlanningLocalMotion($s(q_{i,p}), s(q_i)$)
1 $(\Gamma_i^0, \mathcal{U}_i^0, t_{\Gamma_i^0}) \leftarrow (\{s(q_{i,p})\}, \emptyset, 0)$;
2 Insert($(\Gamma_i^0, \mathcal{U}_i^0, t_{\Gamma_i^0}), LT$);
3 $k \leftarrow 0$;
4 **dowhile** ($\neg$ Empty(LT))
5 Failure $\leftarrow$ False;
6 $s_{cur} \leftarrow$ SelectBestState(LT);
7 $\mathcal{P}_i^k \leftarrow$ ComputePath(q_{cur}, q_i, ϕ_{cur});
8 **if** ($\mathcal{P}_i^k = \emptyset$)
9 *Let* Γ_i^j *be the trajectory ending at* s_{cur};
10 Remove($(\Gamma_i^j, \mathcal{U}_i^j, t_{\Gamma_i^j}), LT$);
11 **endif**;
12 **dowhile** (($\neg$ Failure) $\wedge$ ($\mathcal{P}_i^k \neq \emptyset$))
13 **if** (Convergence($s_{cur}, s(q_i)$))
14 Insert($(\Gamma_i^k, \mathcal{U}_i^k, t_{\Gamma_i^k}), LT$);
15 $(\Gamma_i, \mathcal{U}_i, \Delta T_{\Gamma_i}) \leftarrow$ ExtractTrajectory(LT);
16 **return**$(\Gamma_i, \mathcal{U}_i, \Delta T_{\Gamma_i})$;
17 **endif**
18 $\tilde{\omega} \leftarrow$ TrackingVelocity $(\mathcal{P}_i^k, s_{cur}, \phi_{cur}, s(q_i))$;
19 **for** $(r = 1..NumberWheels)$ **do**
20 ComputeWheelInteractions $(F_{r,\mathcal{A}/\mathcal{T}}, T_{r,\mathcal{A}/\mathcal{T}})$;
21 $U_r(t_{\Gamma_i^k}) \leftarrow$ ComputeControl$(\tilde{\omega}_r, T_{r,\mathcal{A}/\mathcal{T}})$;
22 **endfor**
23 ComputeInternalInteractions($\mathcal{A}$,$\mathcal{T}$);
24 $s_{cur} \leftarrow$ SolveMotion$(s_{cur}, \delta t, \mathcal{A}, U(t_{\Gamma_i^k}), \mathcal{T})$;
25 Failure $\leftarrow$ FailureChecking$(s_{cur}, q_i, \mathcal{P}_i^k)$;
26 **if** ($\neg$ Failure)
27 $(\Gamma_i^k, \mathcal{U}_i^k) \leftarrow (\Gamma_i^k, \mathcal{U}_i^k) \cup (\{s_{cur}\}, \{U(t_{\Gamma_i^k})\})$;
28 $t_{\Gamma_i^k} \leftarrow t_{\Gamma_i^k} + \delta t$;
29 **else**
30 Insert($(\Gamma_i^k, \mathcal{U}_i^k, t_{\Gamma_i^k}), LT$);
31 **endif**;
32 **enddo**;
33 $k \leftarrow k + 1$;
34 **enddo**
35 **return**(Null);
endalgorithm;

A. Generation of $\mathcal{P}_i^k$ (line 7): In the current implementation of the algorithm, $\mathcal{P}_i^k$ is constructed using a technique derived from the Dubins' approach [6]. The obtained sub-path $\mathcal{P}_i^k$ connecting q_{cur} to q_i is a curve without maneuvering points of the form C_1SC_2 where S is a straight line segment, and C_1 and C_2 two circular arcs of radius equal to $\rho_{\phi_{cur}} = L/\tan(\phi_{cur})$ and $\rho_{\phi_{max}}$, respectively, where ϕ_{cur} is the current steering angle of $\mathcal{A}$.

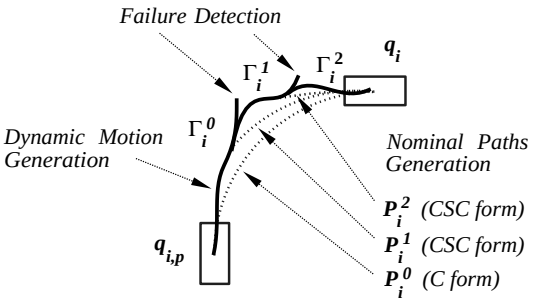

Figure 6: The dynamic motion planning scheme.

B. The tracking function (line 18 to 24): At each time $t_{\Gamma_i^k}$, the angular velocity vector $\tilde{\omega}$ of the wheels required to track $\mathcal{P}_i^k$ is computed from the desired linear and steering speeds $\tilde{v}$ and $\dot{\tilde{\theta}}$ corresponding to the reference point of $\mathcal{A}$. Afterwards, $\tilde{\omega}$ is converted into the control vector U to be applied on $\mathcal{A}$ using the inverse dynamics of the wheels when considering their angular velocity ω at time $t_{\Gamma_i^k} - \delta t$ and the interaction torque vector $T_{\mathcal{A}/T}$. If there exists a control U_r which is not within the admissible interval $[-U_{max}, U_{max}]$, we replace it by U_{max} or $-U_{max}$ according to the sign of the computed U_r. We give further down $\tilde{\omega}$ when $\mathcal{A}$ is executing a sub-path of $\mathcal{P}_i^k$ corresponding to a left forward gyration. For a given axle of $\mathcal{A}$, if $\tilde{v}_M$ is the velocity of its middle point, deduced from $\tilde{v}$ and $\dot{\tilde{\theta}}$, the velocities of its left and right wheels are:

$$\begin{cases} \tilde{\omega}_{left} = \tilde{v}_M(L - l\sin(\phi_{cur}))/(LR_{Wh}\sin(\phi_{cur})) \\ \tilde{\omega}_{right} = \tilde{v}_M(L + l\sin(\phi_{cur}))/(LR_{Wh}\sin(\phi_{cur})) \end{cases}$$

where l is the half of the width of the axles of $\mathcal{A}$, and R_{Wh} is the radius of the wheels. If $\phi_{cur} = 0$, $\tilde{\omega}_{left} = \tilde{\omega}_{right} = \tilde{v}_M/R_{Wh}$.

After having computed the control vector U and the interactions $(F_{\mathcal{A}/\mathcal{T}}, T_{\mathcal{A}/\mathcal{T}})$ of the wheels, we process the internal forces and/or torques applied on $\mathcal{T}$ and on the remaining components of $\mathcal{A}$ as presented in §3.2 and §3.1. The equations of motion of $\mathcal{A}$ and $\mathcal{T}$ are then solved using the time increment δt and the computed forces/torques and the control U.

C. Failure checking (line 25): A failure is detected when $\mathcal{A}$ violates the execution constraints of $\mathcal{EC}$ or when it moves away from the nominal path $\mathcal{P}_i^k$. The first case may occur following an important sliding or skidding of the wheels on $\mathcal{T}$, the violation of the contact relationship between $\mathcal{A}$ and $\mathcal{T}$ —for instance when several wheels are not in contact with $\mathcal{T}$—, or when $\mathcal{A}$ is locked on $\mathcal{T}$—when $\mathcal{A}$ does not move even the angular velocities ω of its wheels are non null.

Since the generation of $\mathcal{P}_i^k$ has not accounted the dynamics of the task, the configuration $q_{cur}(t_{\Gamma_i^k})$ obtained after having applied n successive controls U may be located far from $\mathcal{P}_i^k$. This is considered as failure by the algorithm, and is detected if the nominal path $\mathcal{P}_{cur}$ connecting q_{cur} and q_i is longer and of different form than $\mathcal{P}_i^k$.

5.3 Convergence of the Algorithm

At each time increment δt, the algorithm checks if $\mathcal{A}$ is located at a configuration $q_{i,r}$ in the neighborhood of q_i and verifies the final conditions defined in §5.1. In that case the motion solution and the corresponding controls $(\Gamma_i, \mathcal{U}_i, \Delta T_{\Gamma_i})$ are deduced from the list of the computed sub-solutions $(\Gamma_i^j, \mathcal{U}_i^j, t_{\Gamma_i^j})$ of LT and returned to the geometric level. The node $N(q_i)$ initially corresponding to q_i is then expanded in the search graph $\mathcal{G}$ starting from the reached configuration $q_{i,r}$. If the list LT is empty, q_i is considered as physically unreachable and the corresponding node $N(q_i)$ is not expanded.

In order to reduce the processing time of the local physical planner and to account for the considered global cost function (the length of $\Gamma(q_{start}, q_{goal})$), only shortest sub-paths $\mathcal{P}_i^k$ having a length upper-bounded by an arbitrary value $\bar{\ell}_0$ are generated. This allows to avoid the cases when $\mathcal{A}$ has to move far away from q_i before trying to converge towards it. The term $\bar{\ell}_0$ depends on the

distance between $q_{i,p}$ and q_i and the elevation of the terrain between them. For instance, we have considered $\bar{\ell}_0$ twice the length of $\mathcal{P}_i^0$.

6 Simulation Results

We have shown how a two-level technique combined with a physically based model of the task may help in finding safe and executable motions for a rover moving on a $3D$ terrain. This is based on the use of the physical model of the task, and combines a continuous motion technique and a discrete search strategy in order to deal with several non-trivial features such as collision avoidance, kinematics and dynamics of the vehicle, and its physical interactions with the terrain.

The approach has been implemented on a Sun Sparc workstation. Several experiments for a non-holonomic six-wheeled rover $\mathcal{A}$ have been successfully performed in simulation. As we have mentioned in §2, the advantage of the presented planner is that it deals with dynamics at the planning time. Methods operating in two steps: a full geometric path planning step (as in Figure 7) followed by a dynamic trajectory generation step (Figure 8) require to solve several problems related to the starting configuration to consider when re-planning after having detected a failure, and to the length and the optimality of the resulting solution. These problems are implicitly dealt with by the two-level planning process described in this paper. Figure 9 shows the safe and executable motion solution provided by the planner when solving the same task as in Figure 7. Figure 10 shows a solution when the behavior of the robot is not strongly affected by the local skidding of some of its wheels. In Figures 11 and 12, we show a motion with maneuvering points provided by the planner and the corresponding velocity curve.

The processing time of the planning algorithm depends on several criteria: the irregularities and physical properties of the terrain, the start and goal configurations of the robot, the considered time increment δt to solve the equations of motions of the task, the discretization time ΔT_G of the geometric planning level, and the failures checking (such as allowing the sliding/skidding of some wheels while maintaining an admissible behavior of the robot or not). In general, the solutions provided by the planner require between 20 and 50 minutes depending on the above-mentioned criteria.

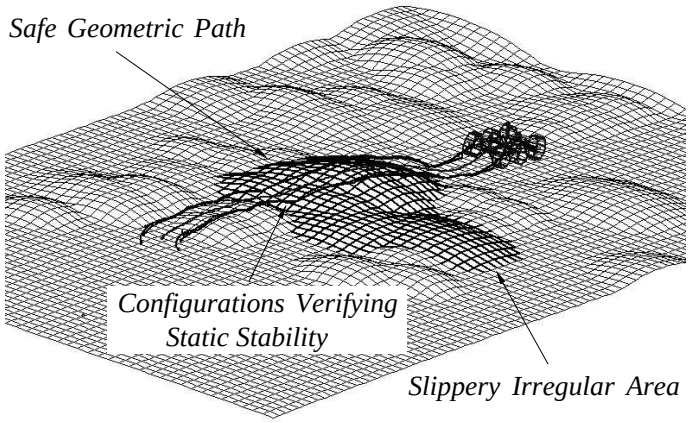

Figure 7: Geometric safe path.

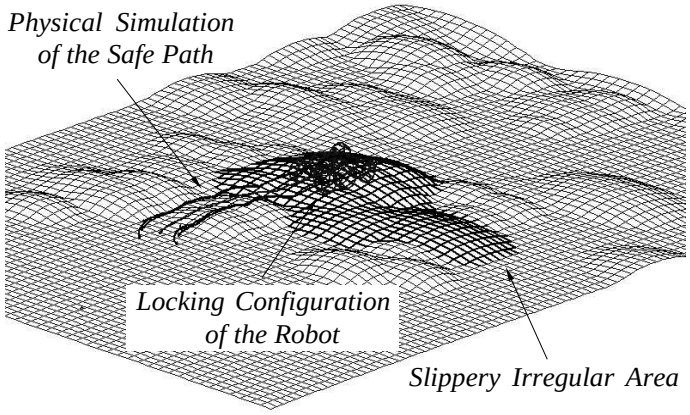

Figure 8: The configuration at which the robot is physically locked.

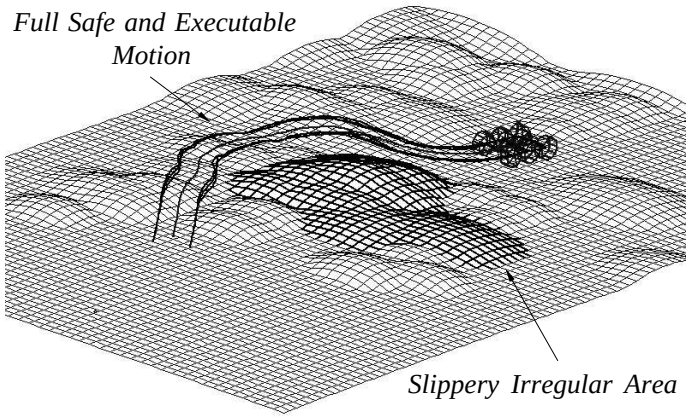

Figure 9: The solution provided by the planner.

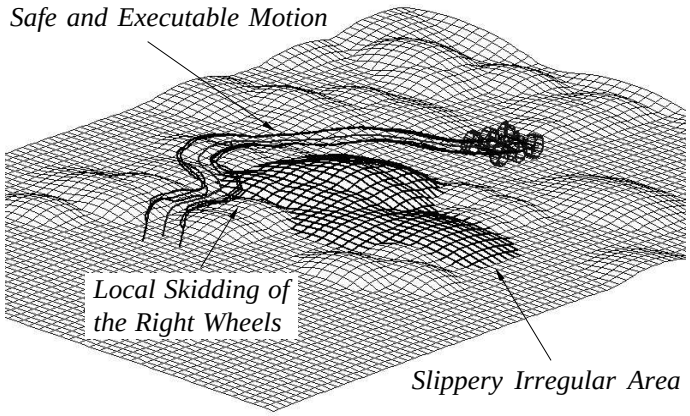

Figure 10: Solution with local skidding of the wheels.

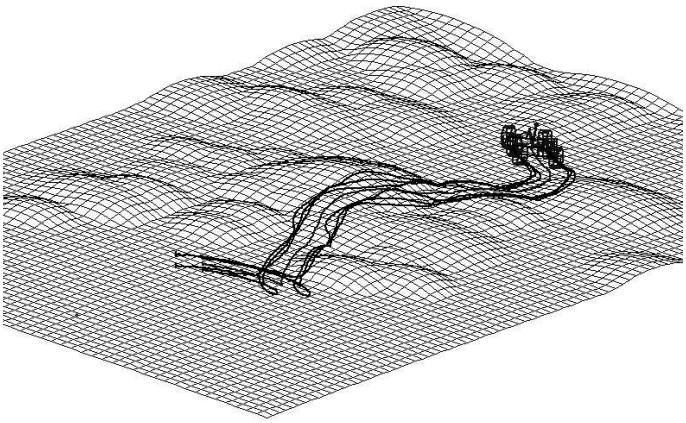

Figure 11: Maneuvering on irregular areas.

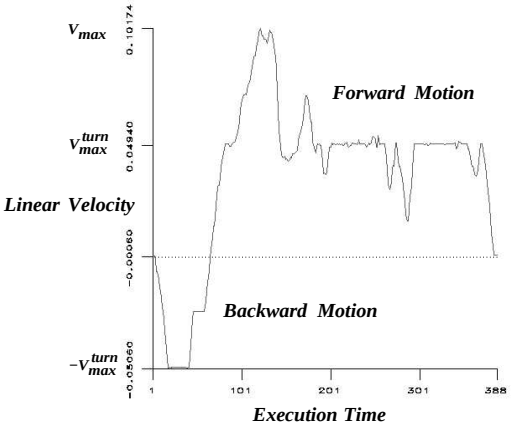

Figure 12: The velocity curve of the robot.

7 Concluding Remarks

We have considered in the presented examples a terrain with no surface mobility. This is only used to demonstrate the approach and to limit the total computation time required to carry out these experiments. In the case of the alteration of the surface of the terrain, several instantiations of the presented physical model of $\mathcal{T}$ have to be used in order to model components such as rocks or movable areas (see [10] for more explanation). Furthermore, the set of the states of the different particles used to represent such components has to be saved and updated each time the algorithm backtracks —which may lead to much computational burden. This is necessary when different local motions located in the same region are searched at different steps of the algorithm. Besides, we assume that the geometric obstacles are of the tail of the vehicle. This allows to make the problem of the collision avoidance more tractable by solving it only at the geometric level when expensing the different nodes of $\mathcal{G}$.

Future work will focus on the automatic construction of the physical models of the terrain, and the identification of the corresponding parameters (i.e. number and distribution of the elementary particles, structure of the corresponding network, and finally the types of interaction between them).

Acknowledgements

This work has been partly supported by the CNES (Centre National des Etudes Spatiales) through the RISP national project and the MRE (Ministère de la Recherche et de l'Espace). It has been also partly supported by the Rhône-Alpes Region through the IMAG/INRIA Robotics project SHARP. The author would like to thank Christian LAUGIER for having supervised this work, and the CROUS at GRENOBLE for its support during its PhD.

References

[1] J. Barraquand, J.C. Latombe, "On Non-Holonomic Mobile Robots and Optimal Maneuvering", *Revue d'Intelligence Artificielle*, 3(2), Hermes, Paris, 1989.

[2] Ch. Bellier, Ch. Laugier, B. Faverjon, "A Kinematic Simulator for Motion Planning of a Mobile Robot on a Terrain", *IEEE/RSJ Int. Conf. on Intelligent Robot and Systems*, Yokohama, Japan, July 1993.

[3] F. Ben Amar, Ph. Bidaud, F. Ben Ouezdou, "On Modeling and Motion Planning of Planetary Vehicles", *IEEE/RSJ Int. Conf. on Intelligent Robot and Systems*, Yokohama, Japan, July 1993.

[4] R. Chatila *et Al.*, "Autonomous Navigation in Natural Environment", *Third Int. Symposium on Experimental Robotics*, Kyoto, Japan, October 1993.

[5] M. Cherif, Ch. Laugier, Ch. Milési-Bellier, B. Faverjon, "Planning the Motions of an All-Terrain Vehicle by Using Geometric and Physical Models", *IEEE Int. Conf. on Rob. Autom.*, San Diego, May 1994.

[6] L.E. Dubins, "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents", *in American Journal of Mathematics*, 79:497-516, 1957.

[7] D. Gaw, A. Meystel, "Minimum-time Navigation of an Unmanned Mobile Robot in a 2-1/2D World with Obstacles", *IEEE Int. Conf. on Rob. Autom.*, San Francisco, April 1986.

[8] H. Goldstein, *Classical Mechanics*, 2nd edition, Addison-Wesley, 1983.

[9] J.E. Hopcroft, J.T. Schwartz, M. Sharir, "Efficient Detection of Intersections among Spheres", *Int. Journal on Robotics Research*, 2(4), 1983.

[10] S. Jimenez, A. Luciani, C. Laugier, "Physical Modeling as an Help for Planning the Motions of a Land Vehicle", *IEEE/RSJ Int. Workshop on Intelligent Robot and Systems*, Osaka, Japan, November 1991.

[11] J.C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, 1991.

[12] J.P. Laumond, P. Jacobs, M. Taix, R. Murray, "A Motion Planner for Non-Holonomic Mobile Robot", LAAS-CNRS Report 92413, Toulouse, October 1992.

[13] A. Luciani and al, "An Unified View of Multiple Behaviour, Flexibility, Plasticity and Fractures: Balls, Bubbles and Agglomerates", *IFIP WG 5.10 on Modeling in Computer Graphics*, Springer Verlag, 1991.

[14] B. Mirtich, J. Canny, "Using Skeletons for Non-Holonomic Path Planning Among Obstacles", *IEEE Int. Conf. on Rob. Autom.*, Nice, France, May 1992.

[15] J. A. Reeds, L. A. Shepp, "Optimal Paths for a Car That Goes Both Forwards and Backwards", *Pacific Journal of Mathematics*, 145(2), 1990

[16] Z. Shiller, Y.R. Gwo, "Dynamic Motion Planning of Autonomous Vehicles", *IEEE Trans. on Rob. Autom.*, vol 7, No 2, April 1991.

[17] Th. Siméon, B. Dacre Wright, "A Practical Motion Planner for All-terrain Mobile Robots", *IEEE/RSJ Int. Conf. on Intelligent Robot and Systems*, Yokohama, Japan, July 1993.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LES NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399